Efficient computation of genotype probabilities for loci with many alleles: II. Iterative method for large, complex pedigrees

R. M. Thallman, G. L. Bennett, J. W. Keele and S. M. Kappes

J Anim Sci 2001. 79:34-44.

Efficient computation of genotype probabilities for loci with many alleles: II. Iterative method for large, complex pedigrees

R. M. Thallman[1], G. L. Bennett, J. W. Keele, and S. M. Kappes

USDA, ARS, Roman L. Hruska U.S. Meat Animal Research Center, Clay Center, NE 68933-0166

ABSTRACT: An algorithm for computing genotype probabilities for marker loci with many alleles in large, complex pedigrees with missing marker data is presented. The algorithm can also be used to calculate grandparental origin probabilities, which summarize the segregation pattern and are useful for mapping quantitative trait loci. The algorithm is iterative and is based on peeling on alleles instead of the traditional peeling on genotypes. This makes the algorithm more computationally efficient for loci with many alleles. The algorithm is approximate in pedigrees that contain loops, including loops generated by full sibs. The algorithm has no restrictions on pedigree structure or missing marker phenotypes, although together those factors affect the degree of approximation. In livestock pedigrees with dense marker data, the degree of approximation may be minimal. The algorithm can be used with an incomplete penetrance model for marker loci. Thus, it takes into account the possibility of marker scoring errors and helps to identify them. The algorithm provides a computationally feasible method to analyze genetic marker data in large, complex livestock pedigrees.

Key Words: Genetic Analysis, Genetic Markers, Pedigree, Statistical Genetics

 J. Anim. Sci. 2001. 79:34–44

Introduction

Marker-assisted selection (**MAS**) in livestock populations will be more useful if it can be applied to multigeneration populations with complex structures. It is likely that marker data will be unavailable for many of the individuals in the pedigree.

The method of peeling (Elston and Stewart, 1971; Fernando et al., 1993) can be used to compute genotype probabilities recursively in pedigrees that do not contain loops. An iterative algorithm has been applied to the peeling formulas (iterative peeling) to compute approximate genotype probabilities in large, looped livestock pedigrees for loci with two alleles (van Arendonk et al., 1989; Kerr and Kinghorn, 1996; Wang et al., 1996). However, the computations required are proportional to the number of alleles raised to the sixth or eighth power, depending on pedigree structure.

Thallman et al. (2001) presented an algorithm, referred to as allelic peeling, that uses a different set of recursive formulas that are much less sensitive to the number of alleles and is thus better suited for use with marker loci with many alleles. However, the recursive algorithm used is restricted to simple pedigrees without loops.

The objective of this research was to extend the method of Thallman et al. (2001) to make it feasible for use with marker loci in large, looped livestock pedigrees. This was accomplished by applying the iterative peeling approach (van Arendonk et al., 1989; Kerr and Kinghorn, 1996; Wang et al., 1996) to the recursive formulas for allelic peeling and using a genetic model that accounts for errors in marker data. An additional objective is the summarization of segregation information in a form that can be used directly in QTL analysis or linkage analysis. The work reported here is a step in the development of an algorithm to analyze multiple linked loci simultaneously.

Materials and Methods

Definitions

The notation and recursive formulas for allelic peeling were defined and discussed in detail by Thallman et al. (2001) and will be used and extended in this paper. Parental prior distributions and progeny likelihoods are properties of parent-offspring pairs that are referred to as meioses and are labeled as separate entities (corresponding to the arrows) in the pedigree. For example, the meiosis from parent i to its offspring k is referred to as meiosis ki. The locus to be analyzed is assumed to be a marker locus with A alleles.

[1]Correspondence: P.O. Box 166 (phone: 402-762-4261; fax: 402-762-4173; E-mail: thallman@email.marc.usda.gov).

Received March 9, 2000.
Accepted August 10, 2000.

The scaled progeny likelihood of meiosis id, $\mathbf{L}(id)$, is proportional to the likelihood of phenotypes connected to a dam, d, through her progeny, i, conditional on the allele transmitted from d to i. It is stored in a column vector of length A and is calculated as

$$\mathbf{L}(id) = c_L(id)^{-1} \left\{\mathbf{M}(i) \circ \prod_{t \epsilon \text{progeny}(i)}^{\circ} [0.5 \cdot \mathbf{L}(ti) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(ti)']\right\} \cdot \mathbf{P}(is) \quad [1]$$

where

$$c_L(id) = \boldsymbol{\pi}' \cdot \left\{\mathbf{M}(i) \circ \prod_{t \epsilon \text{progeny}(i)}^{\circ} [0.5 \cdot \mathbf{L}(ti) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(ti)']\right\} \cdot \mathbf{P}(is)$$

The terms $\mathbf{M}(i)$, $\mathbf{P}(is)$, and $\boldsymbol{\pi}$ are the penetrance function, the parental prior distribution from i's sire to i, and the population allele frequency distribution, respectively, and are defined subsequently and in Thallman et al. (2001). The operator $\cdot$ represents standard matrix or scalar multiplication, and the operator $\circ$ represents elementwise multiplication of matrices. The multiple product is elementwise over each of the progeny of i and is eliminated from the formula if i has no progeny. The operator $'$ indicates matrix transposition. The constant, $\mathbf{1}$, is a column vector of length A filled with ones. The scalar, $c_L(id)$, is a scaling factor.

The scaled progeny likelihood of the paternal meiosis from a sire s to its progeny i, $\mathbf{L}(is)$, is defined similarly and is also stored in a column vector of length A, but is calculated as

$$\mathbf{L}(is) = c_L(is)^{-1} \left\{\mathbf{M}(i) \circ \prod_{t \epsilon \text{progeny}(i)}^{\circ} [0.5 \cdot \mathbf{L}(ti) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(ti)']\right\}' \cdot \mathbf{P}(id) \quad [2]$$

where

$$c_L(is) = \boldsymbol{\pi}' \cdot \left\{\mathbf{M}(i) \circ \prod_{t \epsilon \text{progeny}(i)}^{\circ} [0.5 \cdot \mathbf{L}(ti) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(ti)']\right\}' \cdot \mathbf{P}(id)$$

the difference relative to $\mathbf{L}(id)$ being the transposition of the expression in braces.

The parental prior distribution of meiosis ki, $\mathbf{P}(ki)$, is a column vector of length A with elements containing the probability of the allele (a_{ki}) transmitted from i to its progeny, k, conditional on phenotypes connected to k through its parent, i. It is calculated in a form that is slightly modified from Thallman et al. (2001) to facilitate inferences about segregation:

$$\mathbf{P}(ki) = \mathbf{P0}(ki) + \mathbf{P1}(ki) \quad [3]$$

where

$$\mathbf{P0}(ki) = c_P(ki)^{-1} \cdot 0.5 \cdot \mathbf{P}(id) \circ \left(\left\{\mathbf{M}(i) \circ \prod_{\substack{t \epsilon \text{progeny}(i) \\ t \neq k}}^{\circ} [0.5 \cdot \mathbf{L}(ti) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(ti)']\right\} \cdot \mathbf{P}(is)\right)$$

$$\mathbf{P1}(ki) = c_P(ki)^{-1} \cdot 0.5 \cdot \mathbf{P}(is) \circ \left(\left\{\mathbf{M}(i) \circ \prod_{\substack{t \epsilon \text{progeny}(i) \\ t \neq k}}^{\circ} [0.5 \cdot \mathbf{L}(ti) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(ti)']\right\} \cdot \mathbf{P}(id)\right)$$

$$c_P(ki) = \sum \left[0.5 \cdot \mathbf{P}(id) \circ \left(\left\{\mathbf{M}(i) \circ \prod_{\substack{t \epsilon \text{progeny}(i) \\ t \neq k}}^{\circ} [0.5 \cdot \mathbf{L}(ti) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(ti)']\right\} \cdot \mathbf{P}(is)\right)\right.$$
$$\left. + 0.5 \cdot \mathbf{P}(is) \circ \left(\left\{\mathbf{M}(i) \circ \prod_{\substack{t \epsilon \text{progeny}(i) \\ t \neq k}}^{\circ} [0.5 \cdot \mathbf{L}(ti) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(ti)']\right\}' \cdot \mathbf{P}(id)\right)\right]$$

and the summation in the scaling factor, $c_P(ki)$, is over the elements of the vector.

The genotype distribution of individual i, $\mathbf{G}(i)$, is an A × A matrix with elements that contain the probabilities of the possible genotypes of i conditional on all marker phenotypes in the pedigree. It is calculated as

$$\mathbf{G}(i) = c_G(i)^{-1}\,[\mathbf{P}(\mathrm{id}) \cdot \mathbf{P}(is)'] \circ \left\{\mathbf{M}(i) \circ \prod_{t \epsilon \mathrm{progeny}(i)}{}^{\circ} [0.5 \cdot \mathbf{L}(ti) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(ti)']\right\} \quad [4]$$

where

$$c_G(i) = \sum\left([\mathbf{P}(id) \cdot \mathbf{P}(is)'] \circ \left\{\mathbf{M}(i) \circ \prod_{t \epsilon \mathrm{progeny}(i)}{}^{\circ} [0.5 \cdot \mathbf{L}(ti) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(ti)']\right\}\right)$$

and the summation in the scaling factor, $c_G(i)$, is over the elements of the matrix. The rows and columns of $\mathbf{G}(i)$ correspond to the allele i inherited from its dam and sire, respectively. Thus, $\mathbf{G}(i)$ is the distribution of ordered genotypes that specify which allele was inherited from each parent. An unordered genotype specifies which two alleles an individual possesses but does not indicate from which parent each allele was inherited.

Livestock pedigrees often contain loops. A loop occurs when an individual can be connected to itself through two different parents and(or) progeny. Figure 1 illustrates several different types of loops, each of which occurs commonly in livestock pedigrees. As discussed by Thallman et al. (2001), Eq. [1] to [4] depend on the assumption that the phenotypes connected to an individual through each of its parents and progeny form disjoint subsets of the pedigree and are, therefore, independent conditional on the genotype of the individual. If the individual is part of a loop, then the subsets are not disjoint and therefore may not be independent.

Grandparental Origin Probabilities

In QTL mapping and MAS, genetic markers are used to make inferences about the segregation of QTL alleles through the pedigree. In designed resource families with complete marker data, it is typical for these inferences to take the form of classification (following the rules of Mendelian inheritance) of parent-offspring pairs as either fully informative (inferred unambiguously) or completely uninformative. However, in complex pedigrees with incomplete marker data, many of the parent-offspring pairs are partially informative (one allele is more likely to have been inherited than the other, but neither can be inferred unambiguously). This situation arises when there is uncertainty about the genotype of an individual. Therefore, a probabilistic inference about segregation is useful for QTL mapping with incomplete marker data.

These inferences can take the form of grandparental origin (**GPO**) probabilities. Grandparental origin is defined as the property of a meiosis that specifies whether the allele transmitted from the parent (i) to the offspring (k) was inherited from the grandsire or the granddam. The GPO of meiosis ki is represented by h_{ki} and is coded as 1 if the allele is of grandpaternal origin and 0 if it is of grandmaternal origin. It is functionally equivalent to an element of the "inheritance vector" described by Lander and Green (1987). The GPO probability of meiosis ki, $H(ki)$, is defined as the probability that $h_{ki} = 1$, or the probability of grandpaternal origin.

In Figure 2, assuming that phenotypes are observed without error, i inherited allele 2 from his dam, d, and allele 1 from his sire, s. Therefore, $h_{mi} = 0$ and $h_{pi} = 1$, which implies that m and p received different alleles from i. It is not possible to determine with certainty which allele k inherited from i. Therefore, we are interested in the probability distribution of h_{ki}. The probability that $h_{ki} = 1$ is the probability of grandpaternal origin. In this example it is also the probability that k inherited the same allele as p and a different allele than m.

The GPO probabilities could be computed from genotype probabilities using the rules of Mendelian inheritance. However, it is easier to compute the GPO probabilities directly from parental prior distributions and progeny likelihoods. The GPO probability of meiosis ki, $H(ki)$, is the scalar probability that meiosis ki is of grandpaternal origin ($h_{ki} = 1$), conditional on all of the marker phenotypes. It is calculated as

$$H(ki) = c_H(ki)^{-1} \cdot \mathbf{P1}(ki)' \cdot \mathbf{L}(ki) \quad [5]$$

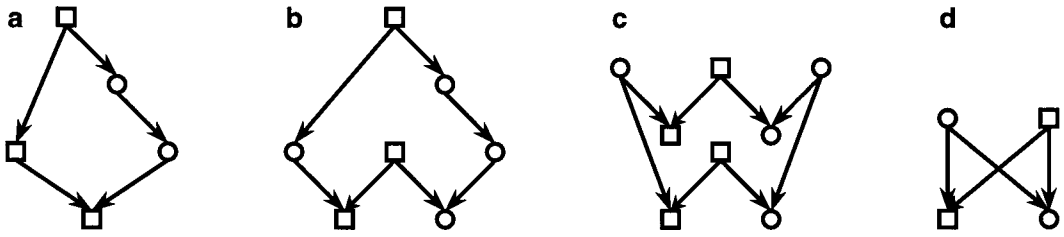

Figure 1. Examples of several types of pedigree loops. (a) An inbreeding loop caused by related parents. (b) A mating loop caused by a sire mated to genetically related females. (c) A mating loop caused by a pair of females being mated to the same sire and then subsequently to a different sire. (d) A mating loop caused by full sibs.

where

$$c_H(ki) = \mathbf{P0}(ki)' \cdot \mathbf{L}(ki) + \mathbf{P1}(ki)' \cdot \mathbf{L}(ki) = \mathbf{P}(ki)' \cdot \mathbf{L}(ki)$$

The GPO probability, H(ki), summarizes the information about h_{ki} that is contained in marker data throughout the pedigree. For example, in Figure 2, H(ki) = 0.71 is the probability that k inherited the allele that i inherited from s.

Computation of GPO probabilities requires the joint parental distribution of the allele transmitted through the meiosis and the GPO of the meiosis. The vectors **P0**(ki) and **P1**(ki) in [3] are the columns of the A × 2 matrix, [**P0**(ki) **P1**(ki)], which contains the joint parental distribution for meiosis ki. Its elements contain the joint probabilities of the allele transmitted in ki with the GPO of ki conditional on all marker phenotypes that are connected to k through i and they sum to one. For example, element 3 of **P1**(ki) contains the joint probability that k inherited the allele that i inherited from its sire *and* that it was the third allele. The marginal parental prior distribution, with respect to only the allele transmitted, **P**(ki), is obtained by summing the two columns. Summing the expressions for **P0**(ki) and **P1**(ki) yields the expression for **P**(ki) in Thallman et al. (2001) and, consequently, the scaling factor, $c_P(ki)$, is identical.

The scalars, **P0**(ki)′ · **L**(ki) and **P1**(ki)′ · **L**(ki), are proportional to the likelihoods of all the marker phenotypes conditional on $h_{ki} = 0$ or 1, respectively. By Bayes theorem, using a prior probability of 0.5 for each of the two states of h_{ki}, the probability of $h_{ki} = 1$ conditional on all phenotypes is the ratio of **P1**(ki)′ · **L**(ki) to its sum with **P0**(ki)′ · **L**(ki), yielding the expression in [5].

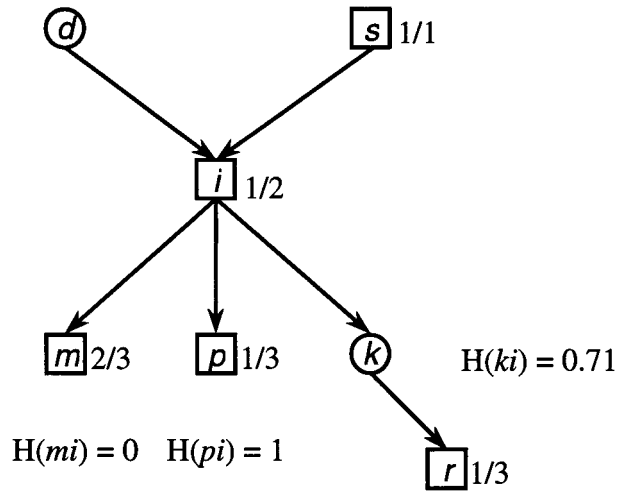

Figure 2. Grandparental origin of three meioses from the same parent (i) in a pedigree with incomplete marker data. Marker phenotypes are indicated to the right of the individuals. Assuming marker phenotypes are observed without error, meioses mi and pi can be inferred to be of grandmaternal ($h_{mi} = 0$) and grandpaternal ($h_{pi} = 1$) origin, respectively. The grandparental origin of meiosis ki can not be inferred unambiguously, but the probability of grandpaternal origin (H(ki) = 0.71) can be computed by allelic peeling.

Incomplete Penetrance

Following Lincoln and Lander (1992), we refer to marker data as phenotypes to emphasize that they are observed with a small degree of error. The penetrance function, **M**(i), is used to relate the genotype (which is unobservable) to the phenotype (assumed herein to be marker data). Specifically, it is the probability distribution of the phenotype conditional on each possible genotype at the locus (i.e., the genetic model). Many computational approaches to analyzing marker data (e.g., Lander and Green [1987]) are dependent on the use of a complete penetrance model, which assumes that marker phenotypes are observed without error. Such methods require error-free marker data.

An incomplete penetrance model incorporates the probability of errors in marker phenotypes and is therefore a more accurate representation of real marker data. An incomplete penetrance model allows the detection of likely scoring errors. Errors in marker data can occur as a result of misinterpretation of electrophoresis results or from mislabeling of samples from collection to extraction to loading on gels.

The simplest form of the penetrance function that allows for errors in marker data assigns a probability of $1 - \varepsilon$ to the phenotype that is consistent with the genotype and distributes the probability of an error (ε) uniformly among all other phenotypes (Ehm et al., 1996). For a codominant marker with A alleles, $\varepsilon_u = \frac{\varepsilon}{A(A+1)/2 - 1}$ is the uniform probability of each erroneous phenotype.

The penetrance matrix for individual i, **M**(i), is an A × A matrix with elements corresponding to the possible genotypes of i (with rows and columns corresponding to the allele i inherited from its dam and sire, respectively). Under the uniform error model, each element takes the value $1 - \varepsilon$ or ε_u, depending on whether the genotype corresponding to the element is consistent or inconsistent, respectively, with the phenotype of i. For example, individual y in Figure 3 has a phenotype of 1/2, so assuming A = 3 and $\varepsilon = 0.01$,

$$\mathbf{M}(y) = \begin{bmatrix} 0.002 & 0.990 & 0.002 \\ 0.990 & 0.002 & 0.002 \\ 0.002 & 0.002 & 0.002 \end{bmatrix}$$

If individual i does not have a phenotype, then **M**(i) is simply a matrix of ones, equivalent to eliminating the term **M**(i) from Eq. [1] to [4]. The elements of this matrix of likelihoods do not sum to one, but instead the sum of the penetrance matrices over all possible phenotypes is a matrix filled with ones.

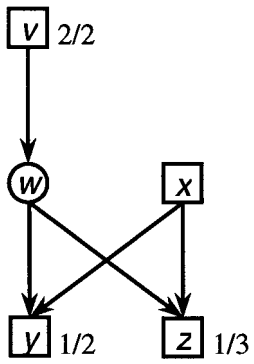

Figure 3. Example pedigree with a full-sib loop.

Iterative Algorithm

Thallman et al. (2001) described a recursive algorithm (allelic peeling) for computing genotype distributions from parental prior distributions and progeny likelihoods. Unfortunately, in complex pedigrees, it often happens that several of the quantities to be computed by the recursive formulas depend on one another, so neither can be computed without first computing the other. This impasse occurs if the pedigree contains a loop and causes the recursive algorithm to fail. Van Arendonk et al. (1989) showed that an approximate solution can be obtained by choosing a starting value for one of the quantities and iterating back and forth between them until they converge. This approach was referred to as "iterative peeling" by Janss et al. (1995). We will describe the application of the iterative algorithm to complex pedigrees using the formulas for allelic peeling.

In iterative allelic peeling, the parental prior distributions and progeny likelihoods are properties of meioses rather than of individuals. Therefore, a list of meioses is constructed, ordered by the birth date of the parent, so that sibs are adjacent to one another. Progeny likelihoods are computed, beginning with the youngest meiosis and proceeding to the oldest, using the population allele frequencies in place of the parental prior distribution. Then, parental prior distributions are computed, beginning with the oldest meiosis and proceeding to the youngest, using the progeny likelihoods just computed. In the subsequent iterations, the computations alternate between parental prior distributions and progeny likelihoods, each computed in terms of the most recent updates of the other.

Progeny likelihoods are computed from youngest to oldest because they are functions of the progeny likelihoods of the progeny as shown in [1] and [2]. The meioses are processed by progeny within parent, accumulating the multiple product. When the last progeny of individual i has been processed, the multiple product is used to compute $\mathbf{L}(is)$ and $\mathbf{L}(id)$, which are stored in an array of progeny likelihoods for random access retrieval. At this point, the multiple product is no longer needed, so there is no need to store this matrix (which has A^2 elements) for each individual in the pedigree.

Parental prior distributions are computed from oldest to youngest because they are functions of the parental prior distributions of the parents as shown in [3]. When a parent i of individual k is not included in the pedigree, then $\mathbf{P}(ki)$ is equal to the vector of population allele frequencies, $\boldsymbol{\pi}$, which has length A.

Then the progeny likelihoods are recalculated using the parental prior distributions just calculated. Iteration continues, always using the most recently calculated parental prior distributions and progeny likelihoods, until the changes in parental prior probabilities from one round to the next are negligible.

Finally, the genotype distribution and GPO probability for each individual and meiosis in the pedigree are computed using [4] and [5], respectively. For a marker with many alleles, it is often adequate to store the GPO probabilities and some summary statistics of $\mathbf{G}(i)$ and discard the full genotypic distributions to save space. Summary statistics that have proven useful are the most likely ordered genotype, the probability of the most likely ordered and unordered genotypes, the total probability of heterozygous genotypes, and the probability of a scoring error.

Also, it is generally useful to permanently store the values of $\mathbf{P}(ki)$, $\mathbf{P1}(ki)$, and $\mathbf{L}(ki)$ at convergence for each meiosis in the pedigree. For loci with many alleles, they require much less storage than the full genotype distributions, and when these values are available it is a trivial process to compute the full genotypic or GPO distribution of any individual on demand.

Computationally Efficient Form of the Multiple Product over Sibs

The expression for the parental prior distribution of the meiosis from parent i to progeny k in [3] includes a multiple product over all of the sibs of k through parent i. Computing this multiple product for each meiosis in the pedigree is the most computationally demanding operation in the peeling algorithm, especially in livestock pedigrees, where sires often have many progeny. In some cases, the efficiency of computing the multiple product can be improved considerably.

When computing parental prior distributions, the meioses are processed by progeny within parent i, accumulating the multiple product over all progeny of i. However, before leaving i, the progeny are processed sequentially a second time in which the multiple product excluding progeny k is computed for each progeny by

$$\prod_{\substack{t\epsilon\text{progeny}(i)\\ t\neq k}}^{\circ} [0.5 \cdot \mathbf{L}(ti) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(ti)'] = \frac{\prod_{t\epsilon\text{progeny}(i)}^{\circ} [0.5 \cdot \mathbf{L}(ti) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(ti)']}{0.5 \cdot \mathbf{L}_a(ki) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}_\mathbf{a}(ki)'} \quad [6]$$

where the division of matrices is elementwise. In other words, the multiple product over the progeny of i excluding k is computed by dividing the multiple product over all the progeny of i by k's contribution to it. Next, the parental prior distribution of meiosis ki is computed, which is the only term that requires the multiple product excluding k. Then, the next progeny of i is processed. Therefore, only one instance of the multiple product over all progeny and one instance of the multiple product excluding a progeny need be stored at a time.

The heuristic form on the left-hand side of [6] is more descriptive of the statistical properties than the computational form on the right-hand side, and if i has only a few progeny there is little difference in computational efficiency. However, if i has 100 progeny, then it is much more efficient to compute the multiple product once and perform the division in [6] 100 times than to compute 100 different multiple products of 99 progeny each, as is indicated by [3].

Equation [6] provides computational savings, but it does require that all elements of the matrix in the denominator be greater than zero. This precludes using it with the complete penetrance model. With the incomplete penetrance model described herein, all genotypes are possible and therefore all elements of the denominator are theoretically greater than zero. However, if k has approximately 100 or more progeny, it is possible for an element of $\mathbf{L}(ki)$ to be so small ($< 10^{-307}$) that it underflows to zero in double precision floating point arithmetic. If this occurs, the form in [3] must be used to compute $\mathbf{P}(ki)$.

Results and Discussion

The term peeling originates from the idea of removing terminal members of a pedigree (individuals that are connected to the pedigree by only one meiosis) by transferring the information from them to their parent or progeny and then repeating the process until there are no remaining members. Iterative peeling (van Arendonk et al., 1989; Janss et al., 1995) transfers information between parents and progeny but does not remove them from the pedigree in the process. Allelic peeling has computational advantages relative to previous peeling algorithms (Thallman et al., 2001).

Iterative allelic peeling combines the advantages of iterative peeling and allelic peeling to address several problems with the application of marker information to livestock pedigrees. Many of the markers available are multi-allelic microsatellites, the pedigrees are large and contain many loops, and there are inevitable errors in marker data.

Iterative allelic peeling also provides a means to summarize information about segregation in the form of grandparental origin probabilities. These probabilities can be used as regression coefficients in within-family QTL analyses. However, they can also be used to quantitatively account for the relationships between families in complex pedigrees. This will be especially important for the application of MAS to livestock populations. Grandparental origin probabilities will be useful for summarizing and transmitting segregation information from one locus to the next in the analysis of multiple linked loci.

Numerical Example

The iterative allelic peeling method is an approximation because it ignores loops, as does an iterative genotypic peeling method suggested by van Arendonk et al. (1989). The extent of the approximation is unknown; however, we show through an example that the approximation can be trivial and we offer additional justification for livestock populations.

Table 1 illustrates the algorithm for iterative allelic peeling of the small looped pedigree in Figure 3. The parental prior distributions and progeny likelihoods were computed by iterating on Eq. [1] through [3] as described above. For example, in the first row of Table 1, $\mathbf{L}(zx)$ is computed by substituting z for i and x for s in Eq. [2]. Because z has no progeny, the multiple product is eliminated from [2] so $\mathbf{L}(zx)$ is equal to $\mathbf{M}(z)' \cdot \mathbf{P}(zw)$ times the scaling factor. The scaling factor is $\boldsymbol{\pi}' \cdot \mathbf{M}(z)' \cdot \mathbf{P}(zw)$. In the first round of iteration, $\boldsymbol{\pi}$ is substituted for $\mathbf{P}(zw)$. The expressions for $\mathbf{L}(yx)$, $\mathbf{L}(zw)$, and $\mathbf{L}(yw)$ are similar. In the expressions for $\mathbf{L}(wv)$, $\mathbf{P}(wv)$, $\mathbf{P}(yx)$, and $\mathbf{P}(zx)$, the parental prior terms are all equal to $\boldsymbol{\pi}$ in all iterations because v and x are founders and therefore, by definition, the parental prior distributions for the meioses from their parents to them are the population allele frequencies. The same is true of the maternal meiosis to w in the expressions for $\mathbf{P}(yw)$ and $\mathbf{P}(zw)$. The penetrance matrices for individuals without phenotypes are omitted from the formulas.

Genotype distributions and GPO probabilities computed from the parental prior distributions and progeny likelihoods at convergence are presented in Table 2. Exact genotype distributions and GPO probabilities were computed directly by summing over the entire joint genotypic distribution of the same pedigree. These exact distributions and the largest absolute differences between iterative allelic peeling and them are presented in Table 2.

Approximation Due to Loops

The example in Figure 3 was designed to accentuate differences due to loops between iterative allelic peeling and the exact method for illustrative purposes. However, iterative allelic peeling produces nearly exact results for many configurations encountered in real data. In experimental data, it is typical for marker data to be collected on one or both of the parents. The dependency in Figure 3 exists because marker data were collected on neither parent. Table 3 contains a comparison of iterative allelic peeling with the exact method for the pedigree in Figure 3 modified by adding a phenotype of 1/3 to parent x. This additional data reduced

Table 1. The iterative allelic peeling algorithm[a] for the example pedigree in Figure 3

Term	Eq.	Calculation omitting scaling factor[b]	First round: Scaling factor	First round: Result	Convergence: Scaling factor	Convergence: Result
$\mathbf{L}(zx)$	[2]	$\mathbf{M}(z)' \cdot \mathbf{P}(zw)$	0.222	$\begin{bmatrix}1.495\\0.009\\1.495\end{bmatrix}$	0.094	$\begin{bmatrix}1.213\\0.021\\1.766\end{bmatrix}$
$\mathbf{L}(yx)$	[2]	$\mathbf{M}(y)' \cdot \mathbf{P}(yw)$	0.222	$\begin{bmatrix}1.495\\1.495\\0.009\end{bmatrix}$	0.204	$\begin{bmatrix}2.392\\0.599\\0.010\end{bmatrix}$
$\mathbf{L}(zw)$	[1]	$\mathbf{M}(z) \cdot \mathbf{P}(zx)$	0.222	$\begin{bmatrix}1.495\\0.009\\1.495\end{bmatrix}$	0.244	$\begin{bmatrix}0.691\\0.008\\2.301\end{bmatrix}$
$\mathbf{L}(yw)$	[1]	$\mathbf{M}(y) \cdot \mathbf{P}(yz)$	0.222	$\begin{bmatrix}1.495\\1.495\\0.009\end{bmatrix}$	0.180	$\begin{bmatrix}0.948\\2.041\\0.011\end{bmatrix}$
$\mathbf{L}(wv)$	[2]	$\{[0.5 \cdot \mathbf{L}(yw) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(yw)'] \circ [0.5 \cdot \mathbf{L}(zw) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(zw)']\} \cdot \boldsymbol{\pi}$	0.877	$\begin{bmatrix}1.705\\0.648\\0.648\end{bmatrix}$	0.616	$\begin{bmatrix}1.025\\0.932\\1.043\end{bmatrix}$
$\mathbf{P}(wv)$	[3]	$0.5 \cdot \boldsymbol{\pi} \circ [\mathbf{M}(v) \cdot \boldsymbol{\pi}] + 0.5 \cdot \boldsymbol{\pi} \circ [\mathbf{M}(v)' \cdot \boldsymbol{\pi}]$	0.112	$\begin{bmatrix}0.006\\0.988\\0.006\end{bmatrix}$	0.112	$\begin{bmatrix}0.006\\0.988\\0.006\end{bmatrix}$
$\mathbf{P}(yw)$	[3]	$0.5 \cdot \boldsymbol{\pi} \circ [[0.5 \cdot \mathbf{L}(zw) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(zw)'] \cdot \mathbf{P}(wv)] + 0.5 \cdot \mathbf{P}(wv) \circ [[0.5 \cdot \mathbf{L}(zw) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(zw)']' \cdot \boldsymbol{\pi}]$	0.513	$\begin{bmatrix}0.254\\0.491\\0.254\end{bmatrix}$	0.513	$\begin{bmatrix}0.121\\0.491\\0.388\end{bmatrix}$
$\mathbf{P}(zw)$	[3]	$0.5 \cdot \boldsymbol{\pi} \circ [[0.5 \cdot \mathbf{L}(yw) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(yw)'] \cdot \mathbf{P}(wv)] + 0.5 \cdot \mathbf{P}(wv) \circ [[0.5 \cdot \mathbf{L}(yw) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(yw)']' \cdot \boldsymbol{\pi}]$	1.243	$\begin{bmatrix}0.203\\0.696\\0.102\end{bmatrix}$	1.511	$\begin{bmatrix}0.166\\0.721\\0.113\end{bmatrix}$
$\mathbf{P}(yx)$	[3]	$0.5 \cdot \boldsymbol{\pi} \circ [[0.5 \cdot \mathbf{L}(zx) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(zx)'] \cdot \boldsymbol{\pi}] + 0.5 \cdot \boldsymbol{\pi} \circ [[0.5 \cdot \mathbf{L}(zx) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(zx)']' \cdot \boldsymbol{\pi}]$	1	$\begin{bmatrix}0.416\\0.168\\0.416\end{bmatrix}$	1	$\begin{bmatrix}0.369\\0.170\\0.461\end{bmatrix}$
$\mathbf{P}(zx)$	[3]	$0.5 \cdot \boldsymbol{\pi} \circ [[0.5 \cdot \mathbf{L}(yx) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(yx)'] \cdot \boldsymbol{\pi}] + 0.5 \cdot \boldsymbol{\pi} \circ [[0.5 \cdot \mathbf{L}(yx) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(yx)']' \cdot \boldsymbol{\pi}]$	1	$\begin{bmatrix}0.416\\0.416\\0.168\end{bmatrix}$	1	$\begin{bmatrix}0.566\\0.266\\0.168\end{bmatrix}$

[a]The terms in the first column are computed using the kernels of the formulas (which are instances of the equations in the second column). The scaling factors are then computed by summing the elements (for parental prior distributions) or multiplying by the vector of allele frequencies (for progeny likelihoods) and the computed values are divided by the scaling factors to obtain the results. Computations proceed from the top of the table to the bottom in each iteration. The results for the first iteration and at covergence are presented. When computing the progeny likelihoods in the first iteration, the parental prior distributions are replaced by $\boldsymbol{\pi}$; thereafter all required quantities are available as indicated in the expressions.

[b]Based on $\boldsymbol{\pi} = \begin{bmatrix}0.333\\0.333\\0.333\end{bmatrix}$, $\mathbf{M}(v) = \begin{bmatrix}0.002 & 0.002 & 0.002\\0.002 & 0.990 & 0.002\\0.002 & 0.002 & 0.002\end{bmatrix}$, $\mathbf{M}(y) = \begin{bmatrix}0.002 & 0.990 & 0.002\\0.990 & 0.002 & 0.002\\0.002 & 0.002 & 0.002\end{bmatrix}$, and $\mathbf{M}(z) = \begin{bmatrix}0.002 & 0.002 & 0.990\\0.002 & 0.002 & 0.002\\0.990 & 0.002 & 0.002\end{bmatrix}$, corresponding to an error rate of $\varepsilon = 0.01$.

the dependency between the parents, *w* and *x*. Using an incomplete penetrance model with the same error rate as in Table 2, the greatest difference in genotype or GPO probabilities between iterative allelic peeling and the exact method was 3×10^{-4}. There was no difference when a complete penetrance model was used.

As shown in the example, full-sib loops can produce a dependency between the genotypes of the two parents, resulting in approximate genotypic and GPO distributions. However, for a dependency to exist between the genotypes of parents, each of the parental genotypes must have at least two plausible states conditional on all the marker data. Otherwise, the joint distribution of the two genotypes would be equal to the product of the two marginal genotype distributions, implying that the two genotypes were independent. Furthermore, the likelihood of full sibs does not depend on the order of the parental genotypes. Therefore, full sibs do not cause

allelic peeling to differ from exact methods when the unordered genotype of either parent is unambiguous.

For example, in Table 3, the ordered genotype of x is ambiguous (it is equally likely to be [1,3] or [3,1]), but the unordered genotype of x is certain to contain alleles 1 and 3 under complete penetrance and almost certain to contain those alleles under incomplete penetrance. Consequently, the dependency is reduced to the extent

Table 2. Genotype and grandparental origin distributions for the example in Figure 3

Term	Calculation omitting scaling factor[a]	Iterative[b] allelic peeling	Exact[c]	δ [d]
$\mathbf{G}(v)$	$[\boldsymbol{\pi} \cdot \boldsymbol{\pi}'] \circ \mathbf{M}(v) \circ [0.5 \cdot \mathbf{L}(wv) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(wv)']$	$\begin{bmatrix} 0.002 & 0.002 & 0.002 \\ 0.002 & 0.983 & 0.002 \\ 0.002 & 0.002 & 0.002 \end{bmatrix}$	$\begin{bmatrix} 0.003 & 0.002 & 0.002 \\ 0.002 & 0.982 & 0.002 \\ 0.002 & 0.002 & 0.002 \end{bmatrix}$	0.001
$\mathbf{G}(w)$	$[\boldsymbol{\pi} \cdot \mathbf{P}(wv)'] \circ [0.5 \cdot \mathbf{L}(yw) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(yw)'] \circ [0.5 \cdot \mathbf{L}(zw) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(zw)']$	$\begin{bmatrix} 0.002 & 0.299 & 0.002 \\ 0.002 & 0.010 & 0.004 \\ 0.002 & 0.678 & 0.000 \end{bmatrix}$	$\begin{bmatrix} 0.004 & 0.331 & 0.002 \\ 0.002 & 0.008 & 0.004 \\ 0.002 & 0.647 & 0.000 \end{bmatrix}$	0.032
$\mathbf{G}(x)$	$[\boldsymbol{\pi} \cdot \boldsymbol{\pi}'] \circ [0.5 \cdot \mathbf{L}(yx) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(yx)'] \circ [0.5 \cdot \mathbf{L}(zx) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(zx)']$	$\begin{bmatrix} 0.326 & 0.104 & 0.201 \\ 0.104 & 0.001 & 0.031 \\ 0.201 & 0.031 & 0.002 \end{bmatrix}$	$\begin{bmatrix} 0.328 & 0.085 & 0.165 \\ 0.085 & 0.001 & 0.085 \\ 0.165 & 0.085 & 0.001 \end{bmatrix}$	0.054
$\mathbf{G}(y)$	$[\mathbf{P}(yw) \cdot \mathbf{P}(yx)'] \circ \mathbf{M}(y)$	$\begin{bmatrix} 0.000 & 0.102 & 0.001 \\ 0.890 & 0.001 & 0.002 \\ 0.001 & 0.001 & 0.002 \end{bmatrix}$	$\begin{bmatrix} 0.000 & 0.173 & 0.001 \\ 0.821 & 0.001 & 0.002 \\ 0.001 & 0.000 & 0.000 \end{bmatrix}$	0.071
$\mathbf{G}(z)$	$[\mathbf{P}(zw) \cdot \mathbf{P}(zx)'] \circ \mathbf{M}(z)$	$\begin{bmatrix} 0.002 & 0.001 & 0.298 \\ 0.009 & 0.004 & 0.003 \\ 0.683 & 0.001 & 0.000 \end{bmatrix}$	$\begin{bmatrix} 0.002 & 0.002 & 0.332 \\ 0.008 & 0.003 & 0.002 \\ 0.650 & 0.000 & 0.000 \end{bmatrix}$	0.034
$\mathbf{P0}(yw)$	$0.5 \cdot \boldsymbol{\pi} \circ [[0.5 \cdot \mathbf{L}(zw) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(zw)'] \cdot \mathbf{P}(wv)]$	$\begin{bmatrix} 0.116 \\ 0.006 \\ 0.378 \end{bmatrix}$	N/A	N/A
$\mathbf{P1}(yw)$	$0.5 \cdot \mathbf{P}(wv) \circ [[0.5 \cdot \mathbf{L}(zw) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(zw)]' \cdot \boldsymbol{\pi}]$	$\begin{bmatrix} 0.005 \\ 0.485 \\ 0.010 \end{bmatrix}$	N/A	N/A
$\mathrm{H}(yw)$[e]	$\dfrac{\mathbf{P1}(yw)' \cdot \mathbf{L}(yw)}{\mathbf{P0}(yw)' \cdot \mathbf{L}(yw) + \mathbf{P1}(yw)' \cdot \mathbf{L}(yw)}$	0.888	0.820	0.068
$\mathbf{P0}(zw)$	$0.5 \cdot \boldsymbol{\pi} \circ [[0.5 \cdot \mathbf{L}(yw) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(yw)'] \cdot \mathbf{P}(wv)]$	$\begin{bmatrix} 0.164 \\ 0.224 \\ 0.112 \end{bmatrix}$	N/A	N/A
$\mathbf{P1}(zw)$	$0.5 \cdot \mathbf{P}(wv) \circ [[0.5 \cdot \mathbf{L}(yw) \cdot \mathbf{1}' + 0.5 \cdot \mathbf{1} \cdot \mathbf{L}(yw)]' \cdot \boldsymbol{\pi}]$	$\begin{bmatrix} 0.002 \\ 0.497 \\ 0.001 \end{bmatrix}$	N/A	N/A
$\mathrm{H}(zw)$	$\dfrac{\mathbf{P1}(zw)' \cdot \mathbf{L}(zw)}{\mathbf{P0}(zw)' \cdot \mathbf{L}(zw) + \mathbf{P1}(zw)' \cdot \mathbf{L}(zw)}$	0.020	0.020	0.000

[a]The terms in the first column are computed using the kernels of the formulas (which are instances of [3] to [5]). Scaling factors are then computed by summing the elements and the kernels are divided by the scaling factors to obtain the results. The kernels of the expressions for $\mathbf{P0}(ki)$ and $\mathbf{P1}(ki)$ are scaled by the scaling factors for $\mathbf{P}(ki)$ listed in Table 1 so that their joint sum is one.

[b]Computed using the expressions in the previous column, the results at convergence in Table 1, and $\boldsymbol{\pi} = \begin{bmatrix} 0.333 \\ 0.333 \\ 0.333 \end{bmatrix}$, $\mathbf{M}(v) = \begin{bmatrix} 0.002 & 0.002 & 0.002 \\ 0.002 & 0.990 & 0.002 \\ 0.002 & 0.002 & 0.002 \end{bmatrix}$, $\mathbf{M}(y) = \begin{bmatrix} 0.002 & 0.990 & 0.002 \\ 0.990 & 0.002 & 0.002 \\ 0.002 & 0.002 & 0.002 \end{bmatrix}$, and $\mathbf{M}(z) = \begin{bmatrix} 0.002 & 0.002 & 0.990 \\ 0.002 & 0.002 & 0.002 \\ 0.990 & 0.002 & 0.002 \end{bmatrix}$, corresponding to an error rate of $\varepsilon = 0.01$.

[c]Exact results were obtained by summing over the entire joint genotypic distribution of the pedigree. Because this method does not depend on partitioning the pedigree into subsets, parental prior distributions were not computed.

[d]Largest absolute difference between corresponding elements of the result from iterative allelic peeling and the exact result.

[e]Results for $\mathrm{H}(wv)$, $\mathrm{H}(yx)$, and $\mathrm{H}(zx)$ are not presented because the parent in each of these meioses is a founder so there is no basis to distinguish between the grandparents. Consequently, $\mathbf{P1}(wv) = \mathbf{P0}(wv) = 0.5 \cdot \mathbf{P}(wv)$ and therefore $\mathrm{H}(wv) = 0.5$ is uninformative by definition. The same is true of $\mathrm{H}(yx)$ and $\mathrm{H}(zx)$.

Table 3. Genotype and grandparental origin distributions for the example in Figure 3 modified to include a phenotype of 1/3 for individual *x*

	Incomplete penetrance[a]		Complete penetrance[b]	
Term	Interative allelic peeling	Difference relative to exact[c]	Iterative allelic peeling	Difference relative to exact[c]
$\mathbf{G}(v)$	$\begin{bmatrix}0.001 & 0.002 & 0.001\\0.002 & 0.990 & 0.002\\0.001 & 0.002 & 0.001\end{bmatrix}$	9×10^{-7}	$\begin{bmatrix}0 & 0 & 0\\0 & 1 & 0\\0 & 0 & 0\end{bmatrix}$	0
$\mathbf{G}(w)$	$\begin{bmatrix}0.000 & 0.492 & 0.000\\0.003 & 0.008 & 0.003\\0.000 & 0.494 & 0.000\end{bmatrix}$	2×10^{-5}	$\begin{bmatrix}0 & 0.5 & 0\\0 & 0 & 0\\0 & 0.5 & 0\end{bmatrix}$	0
$\mathbf{G}(x)$	$\begin{bmatrix}0.002 & 0.001 & 0.498\\0.001 & 0.000 & 0.000\\0.498 & 0.000 & 0.000\end{bmatrix}$	3×10^{-4}	$\begin{bmatrix}0 & 0 & 0.5\\0 & 0 & 0\\0.5 & 0 & 0\end{bmatrix}$	0
$\mathbf{G}(y)$	$\begin{bmatrix}0.001 & 0.001 & 0.001\\0.993 & 0.000 & 0.002\\0.001 & 0.000 & 0.001\end{bmatrix}$	1×10^{-5}	$\begin{bmatrix}0 & 0 & 0\\1 & 0 & 0\\0 & 0 & 0\end{bmatrix}$	0
$\mathbf{G}(z)$	$\begin{bmatrix}0.001 & 0.000 & 0.492\\0.006 & 0.000 & 0.006\\0.494 & 0.000 & 0.001\end{bmatrix}$	2×10^{-5}	$\begin{bmatrix}0 & 0 & 0.5\\0 & 0 & 0\\0.5 & 0 & 0\end{bmatrix}$	0
H(yw)	0.986	2×10^{-5}	1	0
H(zw)	0.014	9×10^{-6}	0	0

[a]Computed using $\boldsymbol{\pi} = \begin{bmatrix}0.333\\0.333\\0.333\end{bmatrix}$, $\mathbf{M}(v) = \begin{bmatrix}0.002 & 0.002 & 0.002\\0.002 & 0.990 & 0.002\\0.002 & 0.002 & 0.002\end{bmatrix}$, $\mathbf{M}(x) = \begin{bmatrix}0.002 & 0.002 & 0.990\\0.002 & 0.002 & 0.002\\0.990 & 0.002 & 0.002\end{bmatrix}$, $\mathbf{M}(y) = \begin{bmatrix}0.002 & 0.990 & 0.002\\0.990 & 0.002 & 0.002\\0.002 & 0.002 & 0.002\end{bmatrix}$, and $\mathbf{M}(z) = \begin{bmatrix}0.002 & 0.002 & 0.990\\0.002 & 0.002 & 0.002\\0.990 & 0.002 & 0.002\end{bmatrix}$, corresponding to an error rate of $\varepsilon = 0.01$.

[b]Computed using $\boldsymbol{\pi} = \begin{bmatrix}0.333\\0.333\\0.333\end{bmatrix}$, $\mathbf{M}(v) = \begin{bmatrix}0 & 0 & 0\\0 & 1 & 0\\0 & 0 & 0\end{bmatrix}$, $\mathbf{M}(x) = \begin{bmatrix}0 & 0 & 1\\0 & 0 & 0\\1 & 0 & 0\end{bmatrix}$, $\mathbf{M}(y) = \begin{bmatrix}0 & 1 & 0\\1 & 0 & 0\\0 & 0 & 0\end{bmatrix}$, and $\mathbf{M}(z) = \begin{bmatrix}0 & 0 & 1\\0 & 0 & 0\\1 & 0 & 0\end{bmatrix}$, corresponding to a compete penetrance model ($\varepsilon = 0$).

[c]Largest absolute difference between corresponding elements of the result from iterative allelic peeling and the exact result, obtained by summing over the entire joint genotypic distribution of the pedigree.

that iterative allelic peeling yields exact or almost exact results.

The dependency in Figure 3 could also have been eliminated by adding the phenotype of *w*, additional full sibs or half sibs, or additional phenotypes connected to the parents through the grandparents. Furthermore, there are a variety of circumstances in which allelic peeling yields exact results despite dependencies between unordered parental genotypes. For example, allelic peeling yields exact results unless at least two of the full sibs have ambiguous ordered genotypes. Although full sibs may cause iterative allelic peeling to yield approximate results as in Table 2, this occurs only in special cases. In most experimental populations used in linkage analysis and QTL mapping, marker phenotypes are collected on the parents and therefore the degree of approximation due to full sibs is trivial.

Other types of loops can also cause approximations in iterative peeling because of dependencies among the genotypes of the individuals in the loops. In order for a loop to cause approximation, the genotype of each individual in the loop must be dependent on the genotypes of the rest of the individuals in the loop. One individual with an unambiguous ordered genotype is sufficient to "break" the loop. Other, less stringent conditions may also be sufficient to break loops.

In many livestock pedigrees, most of the sires have a sufficient number of progeny that their genotypes are known unambiguously. Because every loop includes at least one sire and most loops are longer than those in the example pedigree, the approximation due to loops may cause little loss of information in most livestock pedigrees.

In general, the greatest degree of approximation occurs in parts of the pedigree in which genotypic distributions are ambiguous because there is relatively little marker information. Such parts of the pedigree would usually contribute little to QTL mapping or linkage

analysis even if it was feasible to do an exact analysis. However, adding marker data to these parts of the pedigree can both reduce the degree of approximation and increase the amount of information available.

The approximation for loops in livestock pedigrees could be improved by first identifying which loops have important dependencies, and then applying better approximations locally to those loops. The local approximations could include the cut-extended pedigree method of Wang et al. (1996) or conditioning on the genotypes of a sufficient number of individuals to divide the loop into independent subsets, as in Cannings et al. (1978).

Iterative genotypic peeling (Janss et al., 1995; Kerr and Kinghorn, 1996) handles full-sib loops exactly by summing over the genotypes of both parents, but at substantial computational cost. Other types of loops are approximated the same by iterative genotypic peeling and iterative allelic peeling.

Convergence

Iterative allelic peeling generally converges rapidly but depends on the degree of independence in loops. For Tables 1 and 2, 20 iterations were performed, although parental prior probabilities changed by less than 10^{-3} and 10^{-5} relative to the previous round after the sixth and tenth iterations, respectively. However, in Table 3, convergence occurred much faster because the dependency in the loop was reduced or eliminated. With incomplete penetrance, parental prior probabilities changed by less than 10^{-3} and 10^{-12} relative to the previous round after the third and sixth iterations, respectively, and there were no changes after the second iteration when the complete penetrance model was used. In a pedigree with no loops, iterative allelic peeling converges to exact values in a finite number of iterations.

Incomplete Penetrance

Some computational approaches use a complete penetrance model to enhance computational efficiency by inferring genotypes or limiting the range of possible genotypes of some individuals, assuming no errors in the marker data. The incomplete penetrance model would greatly increase the computational cost of such methods. However, in allelic peeling, the incomplete penetrance model is actually more computationally efficient because [6] can be used with the incomplete penetrance model, but not with complete penetrance. This is especially important in livestock pedigrees in which sires often have a large number of progeny.

The incomplete penetrance function also solves some practical problems that arise when analyzing large, complex pedigrees with incomplete data. The complete penetrance model cannot be used to analyze data that contain "non-Mendelian inheritances" (configurations of phenotypes that are inconsistent with the laws of genetics). These often occur due to marker data errors. Finding the error responsible for a non-Mendelian inheritance can be very tedious in a complex pedigree if complete penetrance is used, but is much easier with incomplete penetrance because the analysis can be performed in spite of the errors and the probability of a data error is computed for each phenotype.

The probability of a scoring error for individual i, $P_{Err}(i)$, is computed as the sum of the probabilities of genotypes that are inconsistent with the phenotype of i. For example, in Table 2,

$$\mathbf{G}(y) = \begin{bmatrix} 0.000 & 0.173 & 0.001 \\ 0.821 & 0.001 & 0.002 \\ 0.001 & 0.000 & 0.000 \end{bmatrix}.$$

and the phenotype of y is 1/2. Therefore,

$$\begin{aligned} P_{Err}(i) &= .000 + .001 + .001 + .002 + .001 + .000 + .000 \\ &= .005. \end{aligned}$$

For simplicity, this paper focuses only on codominant autosomal marker loci. However, it is not difficult to adapt the penetrance function to handle dominant or sex-linked markers or discrete or continuous phenotypes. In each case, the penetrance matrix consists of the likelihood of the observed phenotype conditional on each of the possible genotypes at the locus.

The method of allelic peeling can be applied to looped pedigrees through an iterative algorithm. The resulting probabilities are approximate, but when parents have marker phenotypes, the approximation is minimal. An incomplete penetrance model that accounts for errors in marker data and allows computation of the probabilities of such errors is very useful in complex pedigrees with marker data collected on subsets of the pedigree. The incomplete penetrance function allows greater computational efficiency relative to complete penetrance.

Grandparental origin probabilities condense information from genetic markers into a set of statistics that provides most of the relevant information needed to map QTL based on patterns of genetic segregation. We show for the first time that GPO probabilities can be computed directly as part of a peeling algorithm.

Implications

Iterative allelic peeling is a computationally efficient method for calculating approximate genotype probabilities of loci with many alleles in large pedigrees of arbitrary structure. It also allows calculation of grandparental origin probabilities that indicate the pattern of segregation through the pedigree. Iterative allelic peeling extends the size and complexity of livestock pedigrees that can be used for the detection of quantitative trait loci and for marker-assisted selection. The method described herein is a preliminary step leading toward

an analysis of multiple, linked markers in large, complex livestock pedigrees.

Literature Cited

Cannings, C., E. A. Thompson, and M. H. Skolnick. 1978. Probability functions on complex pedigrees. Adv. Appl. Probab. 10:26–61.

Ehm, M. G., M. Kimmel, and R. W. Cottingham, Jr. 1996. Error detection for genetic data, using likelihood methods. Am. J. Hum. Genet. 58:225–234.

Elston, R. C., and J. Stewart. 1971. A general model for the genetic analysis of pedigree data. Hum. Hered. 21:523–542.

Fernando, R. L., C. Stricker, and R. C. Elston. 1993. An efficient algorithm to compute the posterior genotypic distribution for every member of a pedigree without loops. Theor. Appl. Genet. 87:89–93.

Janss, L. L. G., J. A. M. Van Arendonk, and J. H. J. Van der Werf. 1995. Computing approximate monogenic model likelihoods in large pedigrees with loops. Genet. Sel. Evol. 27:567–579.

Kerr, R. J., and B. P. Kinghorn. 1996. An efficient algorithm for segregation analysis in large populations. J. Anim. Breed. Genet. 113:457–469.

Lander, E. S., and P. Green. 1987. Construction of multilocus genetic linkage maps in humans. Proc. Natl. Acad. Sci. USA 84:2363–2367.

Lincoln, S. E., and E. S. Lander. 1992. Systematic detection of errors in genetic linkage data. Genomics 14:604–610.

Thallman, R. M., G. L. Bennett, J. W. Keele, and S. M. Kappes. 2001. Efficient computation of genotype probabilities for loci with many alleles: I. Allelic peeling. J. Anim. Sci. 79:26–33.

van Arendonk, J. A. M., C. Smith, and B. W. Kennedy. 1989. Method to estimate genotype probabilities at individual loci in farm livestock. Theor. Appl. Genet. 78:735–740.

Wang, T., R. L. Fernando, C. Stricker, and R. C. Elston. 1996. An approximation to the likelihood for a pedigree with loops. Theor. Appl. Genet. 93:1299–1309.